

APPENDIX

The following code listing shows one implementation of the conventional VSCALE routine in accordance with ITU (International Telecommunication Union)-T Recommendation G.728 – Annex G.

5 ;search for maximum positive and negative values in vector

```

10      pcopy y0,y1.      movs.w @r4+,y0
      pcmp x1,y0          movx.w @r4+,x1
      dct pcopy x1,y0
      pcmp x1,y1
      dcf pcopy x1,y1      movx.w @r4+,x1
      pcmp x1,y0
15      dct pcopy x1,y0
      pcmp x1,y1
      dcf pcopy x1,y1      movx.w @r4+,x1
      pcmp x1,y0
      dct pcopy x1,y0
      pcmp x1,y1
20      dcf pcopy x1,y1      movx.w @r4,x1
      pcmp x1,y0          movx.w @r4+r8,x0
      dct pcopy x1,y0      movx.w @r4+r8,x0
      pcmp x1,y1          movx.w @r4+r8,x0
25      dcf pcopy x1,y1      movx.w @r4+r8,x0

```

```

      sts    y0,r1
mov     r1,r0
      sts    y1,r7
30      or     r7,r0
      tst     r0,r0
      bt      VS_ZERO

```

```

35      pabs    y1,y1
      pclr     a0
      pinc     a0,a0
      lds      r6,y0
      psha     #16,y0
      psha     a0,y0,a0

```

```

40      sts     y1,r0
      cmp/ge    r0,r1
      bt/s      vs_pos
      mov       #0,r0

```

```

45      sts     a0, r3
      neg      r3,r3
      mov      r3,r2
      shll     r2
50      cmp/ge    r2,r7
      bf        vsloop3

```

```

      cmp/gt     r7,r3
      bt         vs_end2

```

55 ;Case 3: maximum negative value still has room for normalization

```

      .align     4
vsloop41:
      shal      r7
60      cmp/gt     r7,r3
      bf/s      vsloop41
      add      #1,r0

```

```

    lds    r0,y0
psha #16,y0
    movs.w @r4+,x1
5    psba  x1,y0,a0    movx.w @r4+,x1
    psba  x1,y0,a1    movx.w @r4+,x1
                        movs.w a0,@r5+
                        movx.w a1,@r5+
10    psba  x1,y0,a0    movx.w @r4+,x1
    psba  x1,y0,a1    movx.w @r4+,x1
                        movx.w a0,@r5+
    psba  x1,y0,a0
    movx.w a1,@r5+
                        movx.w a0,@r5+
15
    rts
    nop

;Case 2: maximum negative value exceeds minimum range vsloop3:
20    cmp/ge    r2,r7
    bt     vs_end2
    .align    4
vsloop31:
    shar    r7
25    cmp/ge    r2,r7
bf/s vsloop31
    add     #-1,r0

    lds     r0,y0
30    psba  #16,y0
    movs.w @r4+,x1
    psba  x1,y0,a0    movx.w @r4+,x1
    psba  x1,y0,a1    movx.w @r4+,x1
                        movs.w a0,@r5+
35    movx.w a1,@r5+
    psba  x1,y0,a0    movx.w @r4+,x1
    psba  x1,y0,a1    movx.w @r4+,x1
                        movx.w a0,@r5+
    psba  x1,y0,a0
40    movx.w a1,@r5+
    movx.w a0,@r5+

    rts
    nop
45

;Case 1: zero input vector
VS_ZERO:
    pclr a0
                        movs.w a0,@r5+
50    movx.w a0,@r5+
    movx.w a0,@r5+
    movx.w a0,@r5+
    movx.w a0,@r5+

    mov r6,r0
55    add #1,r0

    rts
    nop

60    .align    4
vs_pos:

```

```

        sts    a0,r2
        mov    r2,r3
add     #-1,r3
        add    r2,r3
5
    cmp/ge    r1,r3
        bf    vsloop5

    cmp/ge    r2,r1
10    bt      vs_end2

;Case 5: maximum positive value still has room for normalization
        .align    4
vsloop61:
15    shal    r1
        cmp/ge    r2,r1
        bf/s    vsloop61
        add    #1,r0
vs_end2:
20    lds     r0,y0
        psha    #16,y0
        movs.w @r4+,x1
        psha    x1,y0,a0    movx.w @r4+,x1
        psha    x1,y0,a1    movx.w @r4+,x1
25    movs.w a0,@r5+
        movx.w a1,@r5+
        psha    x1,y0,a0    movx.w @r4+,x1
        psha    x1,y0,a1    movx.w @r4+,x1
        movx.w a0,@r5+
30    psha    x1,y0,a0
        movx.w a1,@r5+
        movx.w a0,@r5+

        rts
35    nop

;Case 4: maximum positive value exceeds maximum range
vsloop5
40    cmp/ge    r1,r3
        bt      vs_end2

        .align    4
vsloop5:
45    shar    r1
        cmp/ge    r1,r3
        bf/s    vsloop51
        add    #-1,r0

        bra     vs_end2
50    nop

```

The following is an algorithm in accordance with a first embodiment of the present invention.

```

55    ;search for minimum NLS
        movs.w @r4+,x0
        pdmsb x0,a0    movx.w @r4+,x0
        pdmsb x0,y0
60    pcmp a0,y0
        dct pcopy y0,a0    movx.w @r4+,x0

```

```

    pdmsb x0,y0
    pcmp a0,y0
    dct pcopy y0,a0      movx.w @r4+,x0
    pdmsb x0,y0
    pcmp a0,y0
5    dct pcopy y0,a0      movx.w @r4,x0
    pdmsb x0,y0          movx.w @r4+r8,x1;dummy movx to reset r4=&IN[0]
    pcmp a0,y0          movx.w @r4+r8,x1
    dct pcopy y0,a0      movx.w @r4+r8,x1
10    psha #-16,a0        movx.w @r4+r8,x1

    sts a0, r0           ;r0=NLS_MIN

;Case 1: zero input vector
15    cmp/eq #31, r0
    bf/s VSCALE1_check_NLSeq31_end
    mov r6, r7           ;r6=MLS

    mov r6, r0
    add #1, r0           ;set r0=NLS = MLS + 1
    pclr a0

    movs.w a0,@r5+
    movx.w a0,@r5+
    movx.w a0,@r5+
    movx.w a0,@r5+
    movx.w a0,@r5+
25    rts
    nop

30    ;Case 2: non-zero input vector
    VSCALE1_check_NLSeq31_end:
    add #-14, r7         ;r7=MLS-14
    add r7, r0           ;r0=NLS = NLSmin + (MLS-14)
    lds r0, y0
35    psha #16,y0

    movs.w @r4+,x0
    psha x0,y0,a0        movx.w @r4+,x1
    psha x1,y0,a1        movx.w @r4+,x0
    movs.w a0,@r5+
40    psha x0,y0,a0      movx.w a1,@r5+
    movx.w a0,@r5+
    movx.w @r4+,x1
    psha x1,y0,a1        movx.w @r4+,x0
45    psha x0,y0,a0      movx.w a1,@r5+
    movx.w a0,@r5+

    rts
    nop
50

```